GROUP – 54
REPORT ON OOPS PROJECT

Project report submitted in partial fulfilment of the requirements for the completion of the course CS F213 : Object Oriented Programming (OOP)

PROBLEM STATEMENT

Develop an android application (E-MART) which connects customers toretailers and retailers to wholesalers with some listed mandatory functionalities.

GROUP MEMBERS

NAME	ID No.
OJUSTEJA DHOTI	2019A3PS0471H
SHATAKASI GUPTA	2019A3PS0397H
K.HARSHA VARDHAN REDDY	2019AAPS1348H

INTRODUCTION

In this era of pandemic, social distancing and staying at homes has become a must, people have faced immense problems trying to purchase goods which are required everyday. Similarly, the sellers' businesses have also taken a massive hit as the ease of sale of goods has reduced significantly. Physical purchase and selling of goods like old times has become risky and should be avoided as much as possible. Therefore, the system of non-contact home delivery has seen a rise over the last year where people order their requirements online and the goods are delivered to them at their homes fulfilling the needs of customers and the sellers as well.
In this project, we have tried to recreate the same by developing a platform which can be accessed by a customer, a retailer and even a wholesaler as well where they can interact with the each other can carry out their tasks online This app serves as a shopping portal as well as an administrator portal.

Group Member	Role(s)
OJUSTEJA	Login and OTP module
SHATAKSHI	Realtime Firebase, Dashboard, Google Maps and Periodic updating of Status of the items
HARSHA	Review module

BRIEF STUDY

The E-MART consists of the following features:

1.Authentication:

1.1 Description

Provided using firebase authentication provided by Google Cloud which provides various login methods like email, password, phone, etc. In this project, we have used the feature to login by phone number.

1.2 Stimulus/Response

While registering, the user needs to enter his phone number along with other details to create an account.

After registering successfully, to verify the phone number, a onetime password (OTP) is sent to the user's phone number by using the firebase authentication tool.

1.3 Requirements

The user needs to submit the correct phone number and password each time they want to sign in.

2.Notifications:

Toast.class has been whenever a pop-up has to be displayed to the user. This includes cases like, successful/failed attempt to login, added to cart, cancelled order, completed payment, etc.

3.Customer activity:

The app provides the following features to the customer:

- Login and registration.
- Buying items by adding them to cart and finally displaying the bill containing the

items selected, quantity, price and total amount, address of the retailer.

- Providing feedback about their experience with the app.
- Cancellation of order.
- Review of products provided by other customer.

4.Retailer Activity:

The app provides the following features to the retailer:

- Login and registration.
- Buying items in the same way as a customer along with cancellation.
- Providing feedback about their experience with the app.
- Automatic updating of stock after each customer's order.
- Adding items to the database and setting quantity and price.
- Delete items from the database.
- Review orders placed by customers.

5.Wholesaler Activity:

The app provides the following features to the retailer:

- Login and registration.
- Providing feedback about their experience with the app.
- Automatic updating of stock after each retailer's order.
- Adding items to the database and setting quantity and price.
- Delete items from the database,
- Review orders placed by retailers

SOFTWARE REQUIREMENTS

1.Software

Jetbrains Android Studio (version 4.1) has been used for development of the app. The software provides a basis for creating android applications without the need to write things from scratch. The UI development is therefore made a little easier and linking the backend with the UI becomes convenient. We have created the app keeping in mind that many people might not be owning higher end phones. Therefore, the app supports a minimum requirement of Android Marshmallow 6.0 which covers more than 80% of the active smartphones.

2.Language

Java was used for developing the backend of the application which includes all the functionalities performed by the application.
For the frontend, we used xml layouts for designing the UI of each page present in the

app. Features like buttons, text boxes, lists, search bar were added to each page as per the requirement after which they were linked to the backend of the project using the Java Class.

3.Emulator

Debugging or testing an application is difficult without having immediate access to an emulator. Therefore, we have used an inbuilt emulator provided by Android Studio.
Specifications:
● Phone : Google Galaxy Nexus
● Android OS Version : Nougat 7.1
● API level : 254.Firebase Authentication
Firebase Authentication has been used in this project for verifying the user details before signing them and redirecting to their respective dashboards. The dependencies can be set up by Tools > Firebase > Firebase Authentication > Link Project > Set up Dependencies.

5.Firebase Realtime Database

Firebase Realtime Database is another platform service provided by Google Cloud which allows the user to store data in databases programmatically or directly. After setting up dependencies in a way similar to Firebase Authentication, the user can access or store data on the database fields or child fields as per requirement.

6.APIs

Google Maps API was used to obtain location of the user with an autofill search bar used in Google Maps.

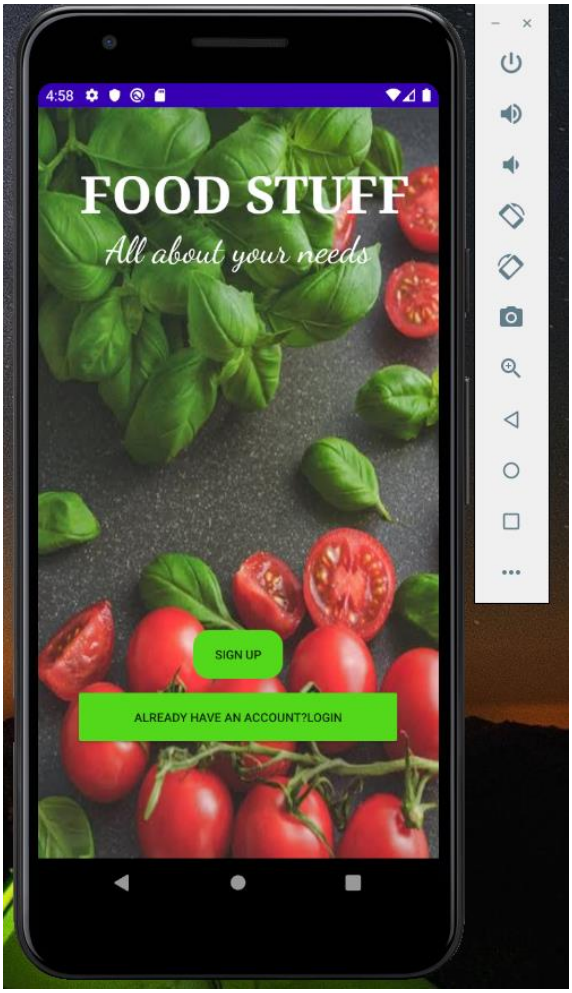
ADDITIONAL FEATURES

As suggested in the problem statement, we were supposed to create options for wholesalers and retailers to manually update the database after each order. Instead, we have created a method which automatically updates the database after order completion thus reducing the job of the sellers.
We have also added the optional feature cancelling the order, where the user can cancel their latest order, the updated quantities in the database will be reverted.

DETAILED WORKING OF THE APPLICATION

Main Activity

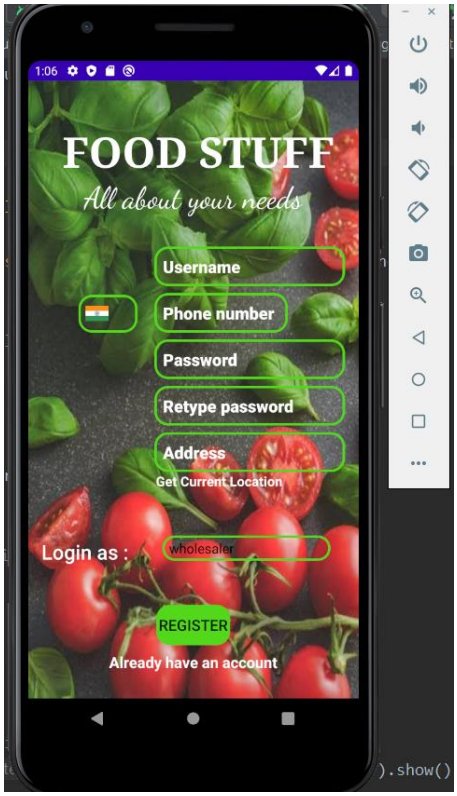
Upon opening the application, the Main Activity is initiated in which an option is given to sign up or login in case if you have already registered.



Sign Up Activity

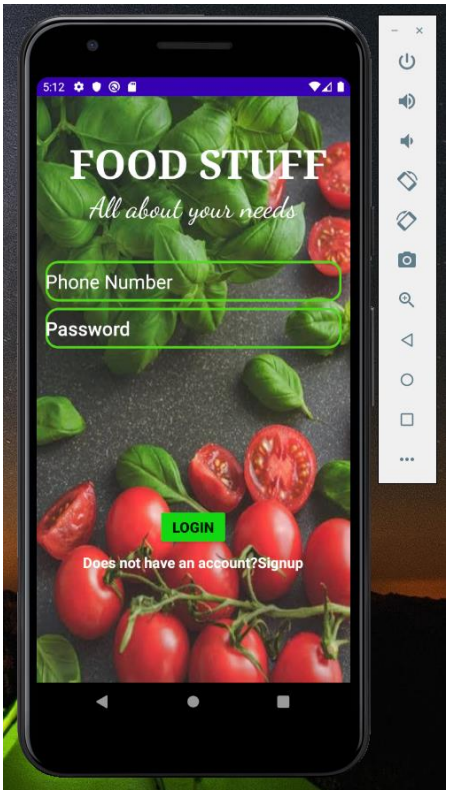
In this activity, the user needs to enter his registered using his phone number. This page contains the fields - username, password, retype password, phone number, address (using Google Maps API) and selecting from one ofthe following options (customer, retailer, wholesaler) using a spiner. If the user is already registered with a phone number, another account cannot be created using the same phone number.Alternatively, they can click on the **(Already have an account**) text to open the login portal.
Firstly the phone number is checked, if the number entered is already registered, signup activity is redirected to login activity with a pop message of "Account already existing " The main purpose of this activity is to feed information of the user into the realtime database.

```
FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference databaseReference = database.getReference().child("profile")
```



Login Activity

If a user already has an account, he can login using this activity by entering his registered phone number and password.
Firebase authentication is used to check whether the user is already registered or not using the fuction **fAuth.getCurrentUser()** and password is used for the safety purposes.

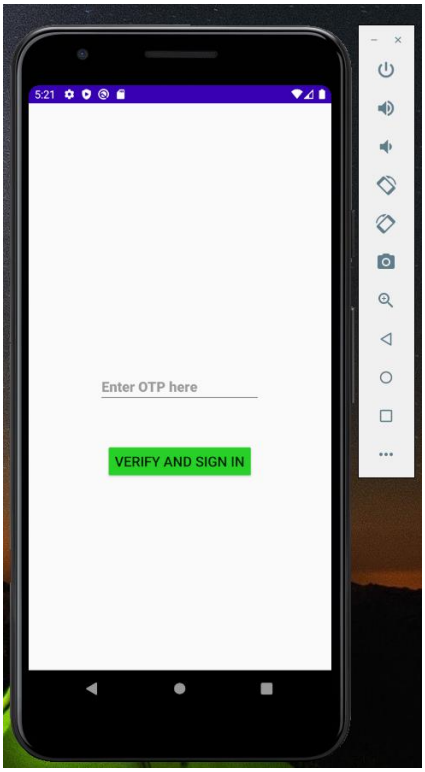


OTP Activity

After the user registers, his phone number is verified by sending an otp to his registered phone number.

```
PhoneAuthOptions options =
        PhoneAuthOptions.newBuilder(mAuth)
                .setPhoneNumber(phoneNumber)       // Phone number to verify
                .setTimeout(60L, TimeUnit.SECONDS) // Timeout and unit
                .setActivity(this)                 // Activity (for callback binding)
                .setCallbacks()
```

The above function is used for request for otp generation.
signInWithPhoneAuthCredential() is used for the verification if the entered otp and the generated is same or not.

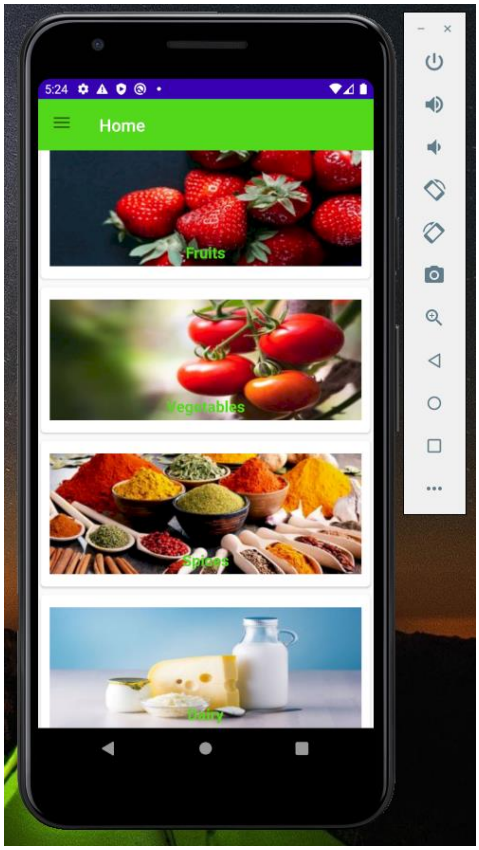


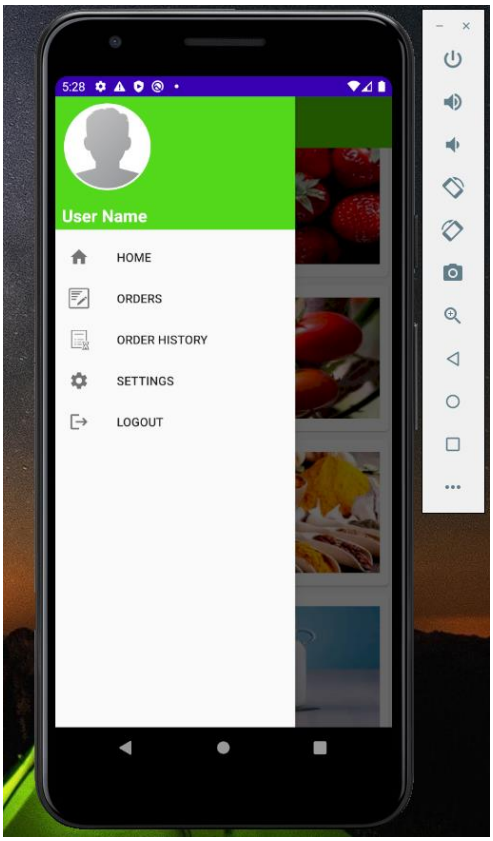
Dashboard

It is the home activity of our application. We have designed it differently for each type of user namely – Wholesaler, Retailer and Customer.
It shows different categories from which the user can shop and navigation drawer layout with different menu items for each user.
Different options in the menu when clicked opens up different fragment like - home fragment, setting fragment, logout (common to all the three users) etc.
Recyclerview is used for displaying of the different categories.

recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));

myadapter = new Adapter(dataqueue(),getContext());
recyclerView.setAdapter(myadapter);



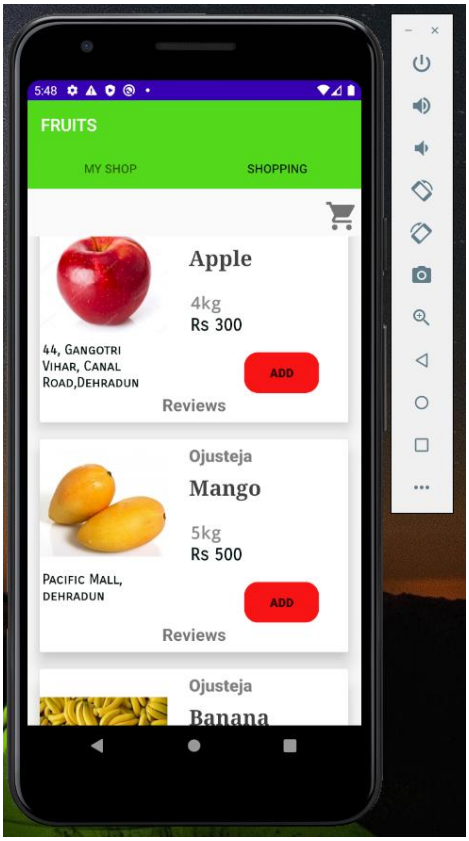


Categories Activity

We had created the theme of "shop by categories" and hence created four categories of – Fruits, Vegetables, Spices and Dairy. Every product has its own review and a user can see it by clicking on the button "Reviews". Then there is "Add to cart" button for adding the products into the cart.
For displaying the products in recycler view , data is retrieved from the firebase.

```
FirebaseRecyclerOptions<ProductModel> options =
        new FirebaseRecyclerOptions.Builder<ProductModel>()

.setQuery(database.getReference("users").child("wholesaler").child("Fruits"),Product
Model.class)
            .build();
```



Cart Activity

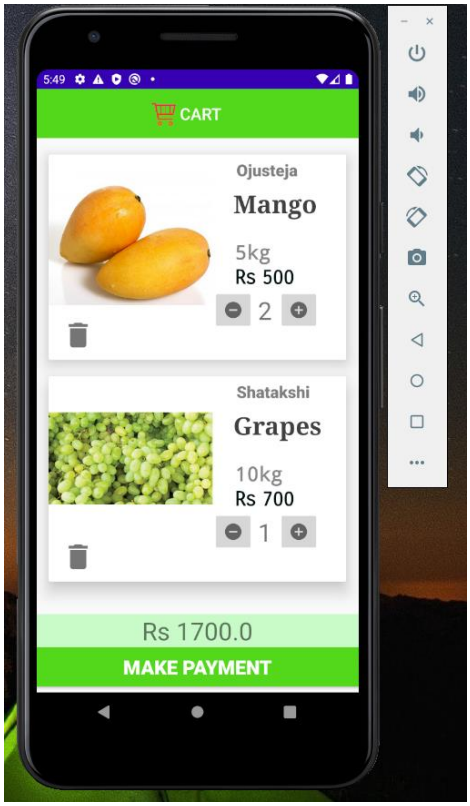
This activity shows all the items which the user has added to its cart. Here the user can the change the number of quantity he wants to buy. Delete option is also provided for the user to remove a particular item from the cart. At the bottom of the page, total price of all the products in the cart is mentioned. The last button of “Make payment” is provided for the final confirmation of the placing of order. After clicking on Make Payment, cart is cleared and order is placed and all the items are moved to Placed Order Activity.
Same as the categories activity , data is retrieved from the firebase.

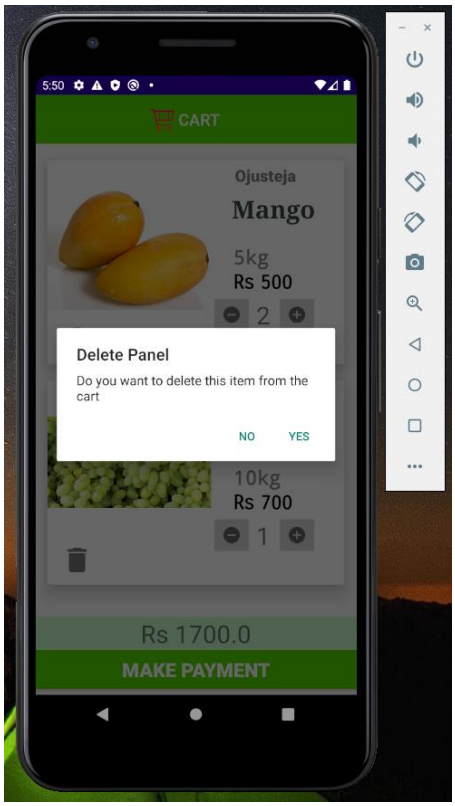
For the pop up dialog box:

```
AlertDialog.Builder builder = new AlertDialog.Builder(holder.price.getContext());
builder.setTitle("Delete Panel");
builder.setMessage("Do you want to delete this item from the cart");
```

Delete option is done by the function:

```
FirebaseDatabase.getInstance().getReference("cart").child(setProfileData.getProfilePhoneNumber())
        .child(getRef(position).getKey()).removeValue();
```





Settings Activity

This activity allows the user to edit this personal information like his - profile image, username, password etc. This time Map is used to edit the information the realtime database.

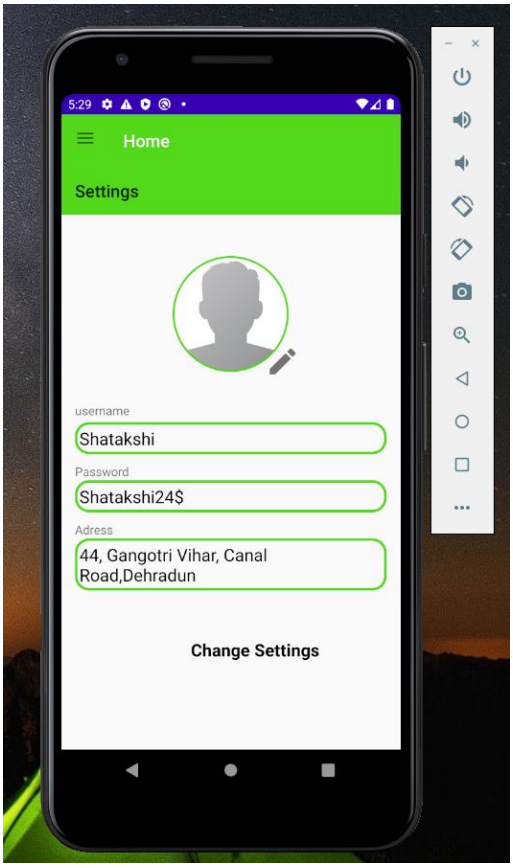
Map<String,Object> map = new HashMap<>();

map.put("username",editUsername .getText().toString());

map.put("password",editPassword.getText().toString());

map.put("address",editAddress.getText().toString());

FirebaseDatabase.getInstance().getReference("profile").child(setProfileData.getProfilePhoneNumber()).updateChildren(map);



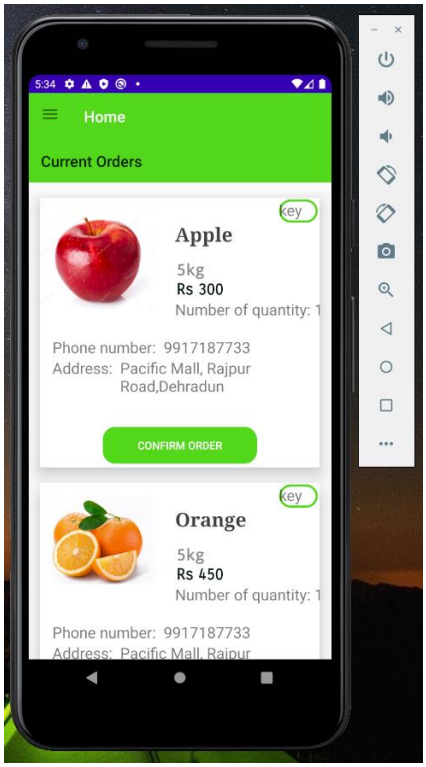
Current Orders Activity

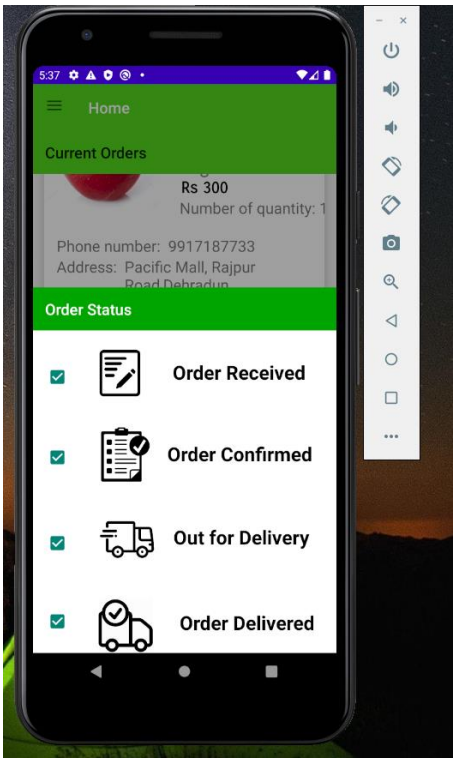
This Activity displays the current orders of the Wholesaler and Retailer and it also allows you to manage the status of the orders. For the status a dialog box appears in which different checkboxes are placed. Retailer/Wholesaler checks the boxed for showing the status of the product.

For creating the dialog box:

Final DialogPlus dialogPlus =

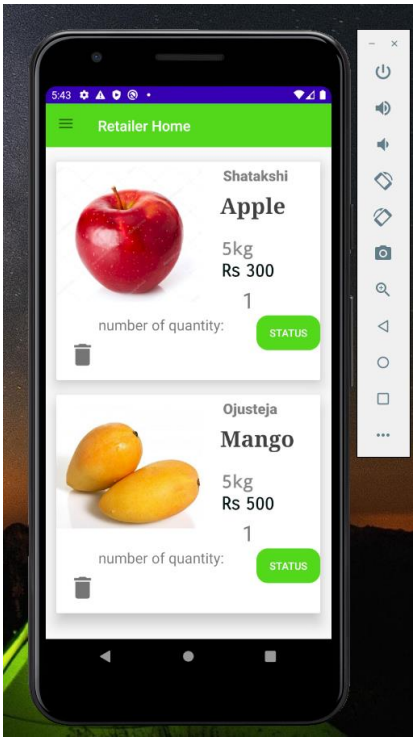
DialogPlus.newDialog(getContext().setContentHolder(new ViewHolder(R.layout.status_content_layout)) . setExpanded(true, 1300) .create();





Order Placed Activity

This Activty shows the items which the user has placed order for and it also shows the status of the products. It also contained the option of cancelling the order.



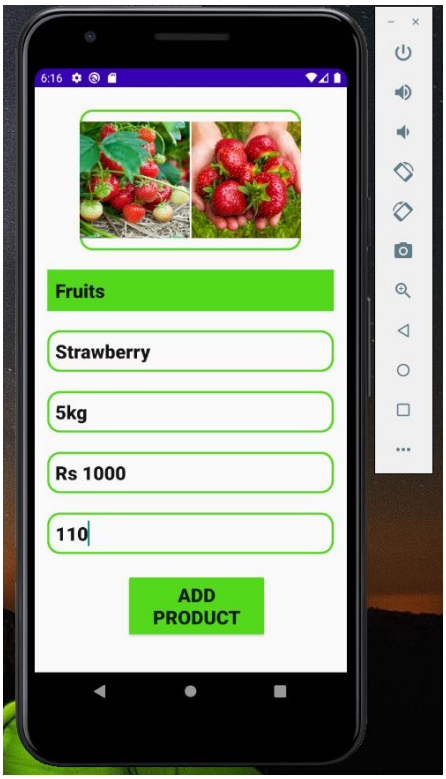
Add Product Activity for wholesaler and Retailer

This activity allows the wholesaler and the retailer to add new products in their shop. Image of the product can also be added and is uploded to the firebase.
Option of uploading the Image is also given.

Dexter.withActivity(AddNewProduct.this)
.withPermission(Manifest.permission.READ_EXTERNAL_STORAGE)

Permission for allowing the access of media.

```
protected void onActivityResult(int requestCode, int resultCode,Intent data)
   {
      super.onActivityResult(requestCode, resultCode, data);
      if (requestCode == 1 && resultCode == RESULT_OK)
      {
         filepath = data.getData();
         try
            {
               InputStream inputStream =
getContentResolver().openInputStream(filepath);
               bitmap = BitmapFactory.decodeStream(inputStream);
               mNewProductImage.setImageBitmap(bitmap);
//showing the image in the box.
```

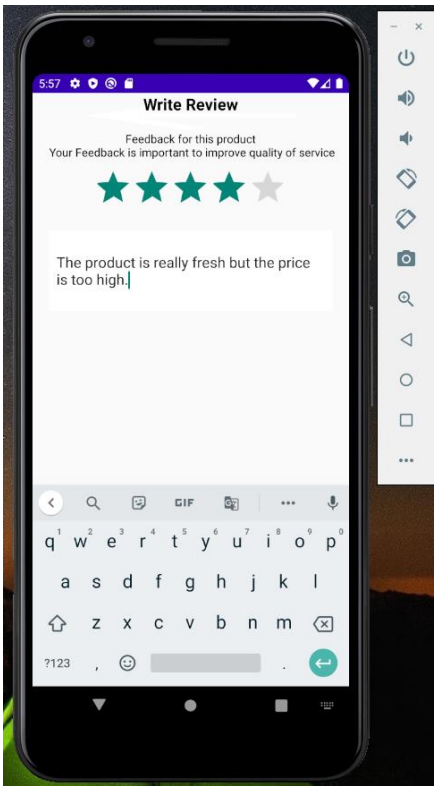


Review Activity

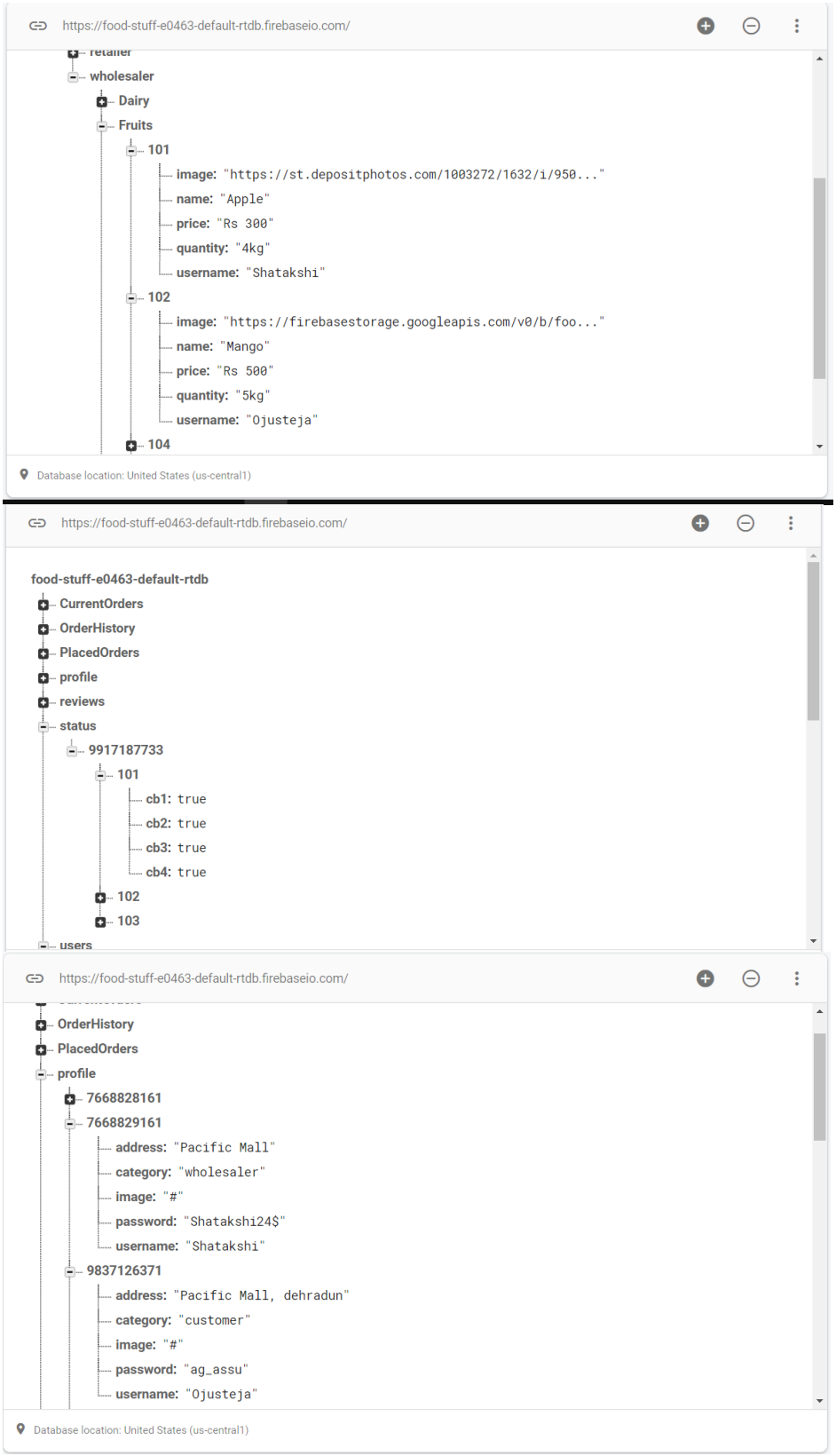
This Activity is for the users to give their feedback and review about a particular product. Review is stored in the firebase and is retrieved by the option Review during the display of the product in the dashboard.

```
FirebaseRecyclerOptions<ReviewModel> options =
        new FirebaseRecyclerOptions.Builder<ReviewModel>()

.setQuery(database.getReference().child("reviews").child(key),ReviewModel.class)
        .build();
```



A look at the realtime database:



PROBLEMS FACED

- Familiarizing ourselves with the working of Android Studio and Firebase.
- Learning the use to link between project and database to retrieve and add information to the database.
- Time constraints.
- Working amidst the problems faced in these difficult times, where we have faced immense stress and fatigue.
- One teammate family members were o tested positive for covid.
- Working together on the same project from 3 different locations.

ACTIONS TAKEN

- Learning all the important concepts well before starting the project.
- Watch videos and read documentations to access databases from the project while trying to keep the code as neat and optimized as possible.
- Peers worked on their own convenient times keeping the deadlines in mind.
- Keep ourselves motivated and worked on each and every problem together.
- Tried to manage the workload accordingly till the person was fit to return.
- Using tools like github or drive to send important information or code files to each other for easy access.

REFERENCES

- https://developer.android.com/docs
- https://developers.google.com/maps/documentation/distance-matrix/overview
- https://firebase.google.com/docs
- https://www.youtube.com/c/MdJamalmca/featured
- https://www.youtube.com/watch?v=kRAyXxgwOhQ&list=PLirRGafa75rRquGkDxwcJ-sBCktcJGV6J